Clustering
Lecture 4: Density-based Methods

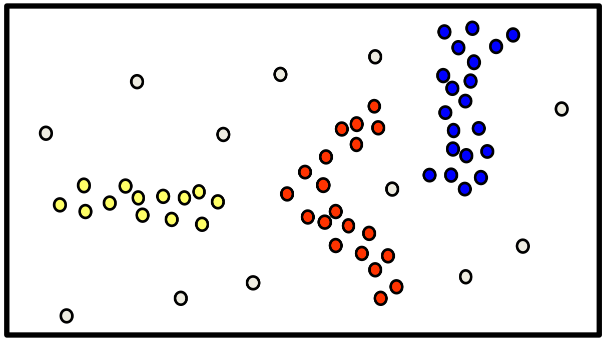
Jing Gao
SUNY Buffalo

Outline

- **Basics**
  - Motivation, definition, evaluation
- **Methods**
  - Partitional
  - Hierarchical
  - Density-based
  - Mixture model
  - Spectral methods
- **Advanced topics**
  - Clustering ensemble
  - Clustering in MapReduce
  - Semi-supervised clustering, subspace clustering, co-clustering, etc.

Density-based Clustering

- **Basic idea**
  - Clusters are dense regions in the data space, separated by regions of lower object density
  - A cluster is defined as a maximal set of density-connected points
  - Discovers clusters of arbitrary shape
- **Method**
  - DBSCAN

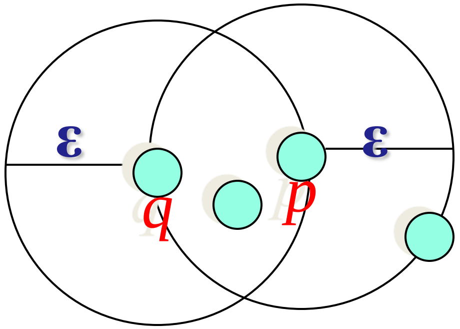


Density Definition

- ε-Neighborhood – Objects within a radius of ε from an object.

$$N_{\varepsilon}(p):\{q \mid d(p,q) \le \varepsilon\}$$

- “High density” - ε-Neighborhood of an object contains at least *MinPts* of objects.

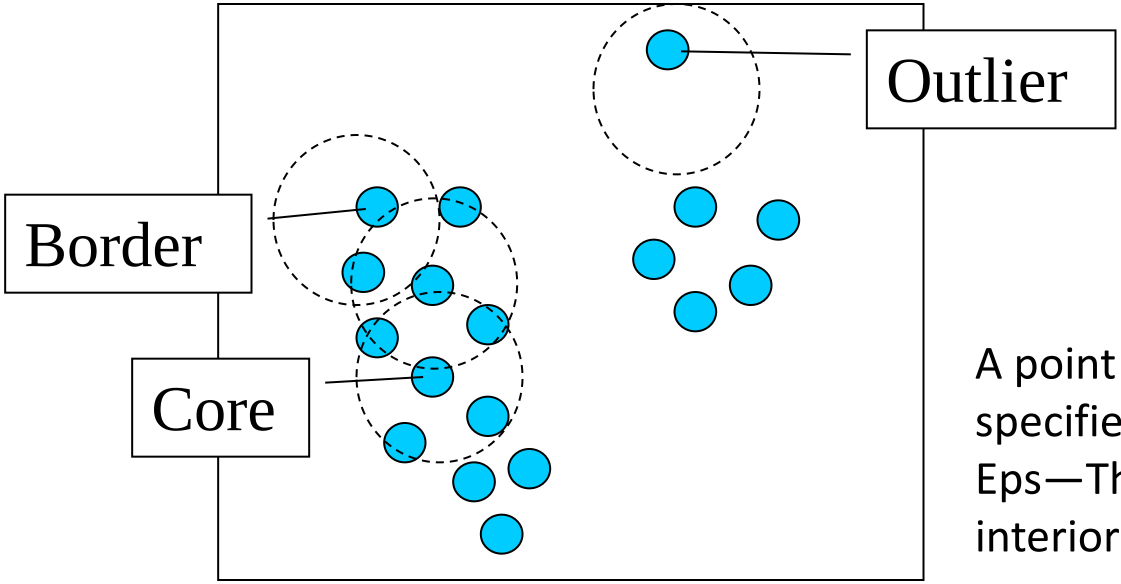


ε-Neighborhood of *p*
ε-Neighborhood of *q*

Density of p is “high” (MinPts = 4)

Density of q is “low” (MinPts = 4)

Core, Border & Outlier



Given *ε* and *MinPts*, categorize the objects into three exclusive groups.

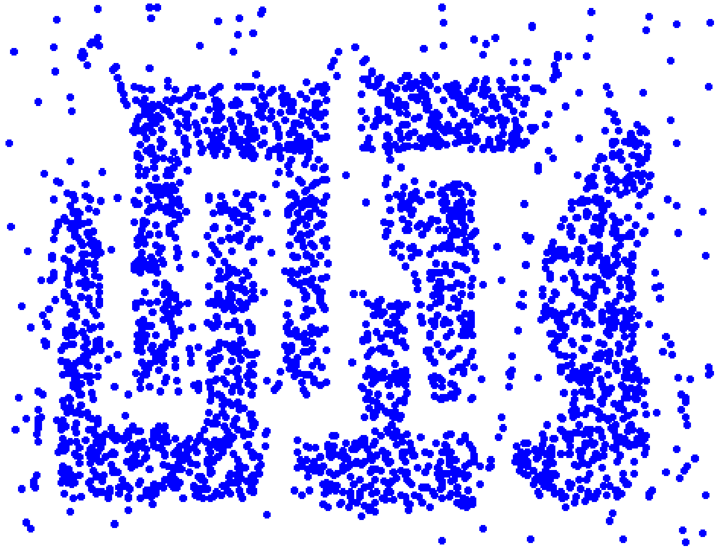
A point is a core point if it has more than a specified number of points (MinPts) within Eps—These are points that are at the interior of a cluster.

ε = 1unit, MinPts = 5

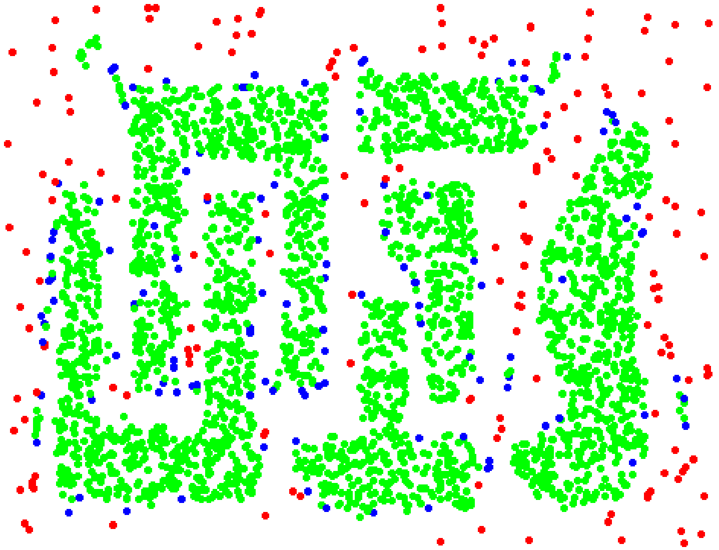
A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.

A noise point is any point that is not a core point nor a border point.

Example



Original Points

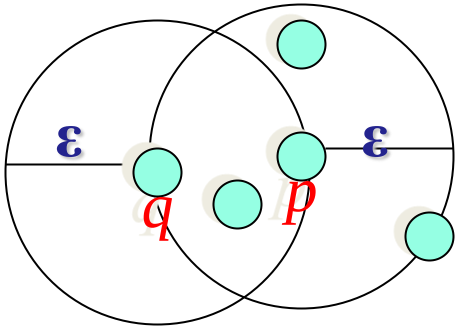


Point types: core, border and outliers

ε = 10, MinPts = 4

Density-reachability

- Directly density-reachable
  - An object *q* is directly density-reachable from object *p* if *p* is a core object and *q* is in p's ε-neighborhood.

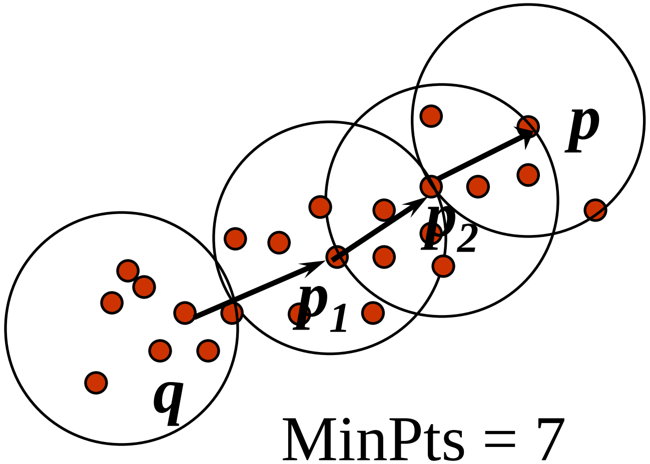


MinPts = 4

- *q* is directly density-reachable from *p*
- *p* is not directly density-reachable from *q*
- Density-reachability is asymmetric

Density-reachability

- Density-Reachable (directly and indirectly):
  - A point *p* is directly density-reachable from p_2
  - p_2 is directly density-reachable from p_1
  - p_1 is directly density-reachable from *q*
  - $p \leftarrow p_2 \leftarrow p_1 \leftarrow q$ form a chain

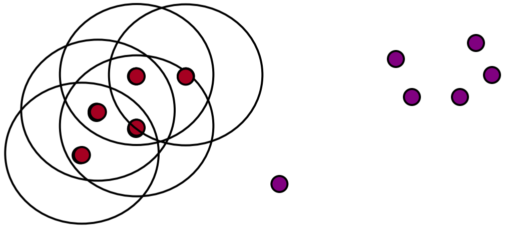


MinPts = 7

- *p* is (indirectly) density-reachable from *q*
- *q* is not density-reachable from *p*

DBSCAN Algorithm: Example

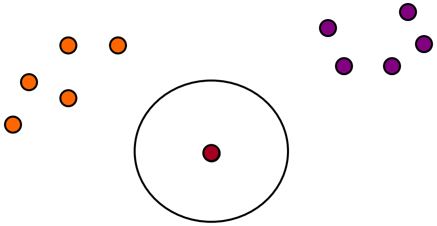
- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



```
for each $o \in D$ do
    if $o$ is not yet classified then
        if $o$ is a core-object then
            collect all objects density-reachable from $o$
            and assign them to a new cluster.
        else
            assign $o$ to NOISE
```

DBSCAN Algorithm: Example

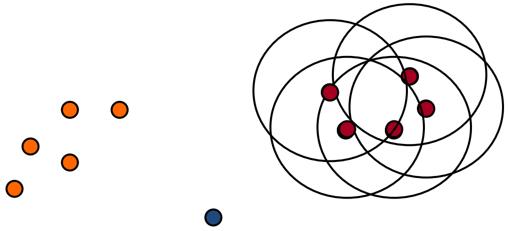
- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



```
for each $o \in D$ do
    if $o$ is not yet classified then
        if $o$ is a core-object then
            collect all objects density-reachable from $o$
            and assign them to a new cluster.
        else
            assign $o$ to NOISE
```

DBSCAN Algorithm: Example

- **Parameter**
    - ε = 2 cm
    - *MinPts* = 3



```
for each $o \in D$ do
    if $o$ is not yet classified then
        if $o$ is a core-object then
            collect all objects density-reachable from $o$
            and assign them to a new cluster.
        else
            assign $o$ to NOISE
```

DBSCAN Algorithm: Pseudocode

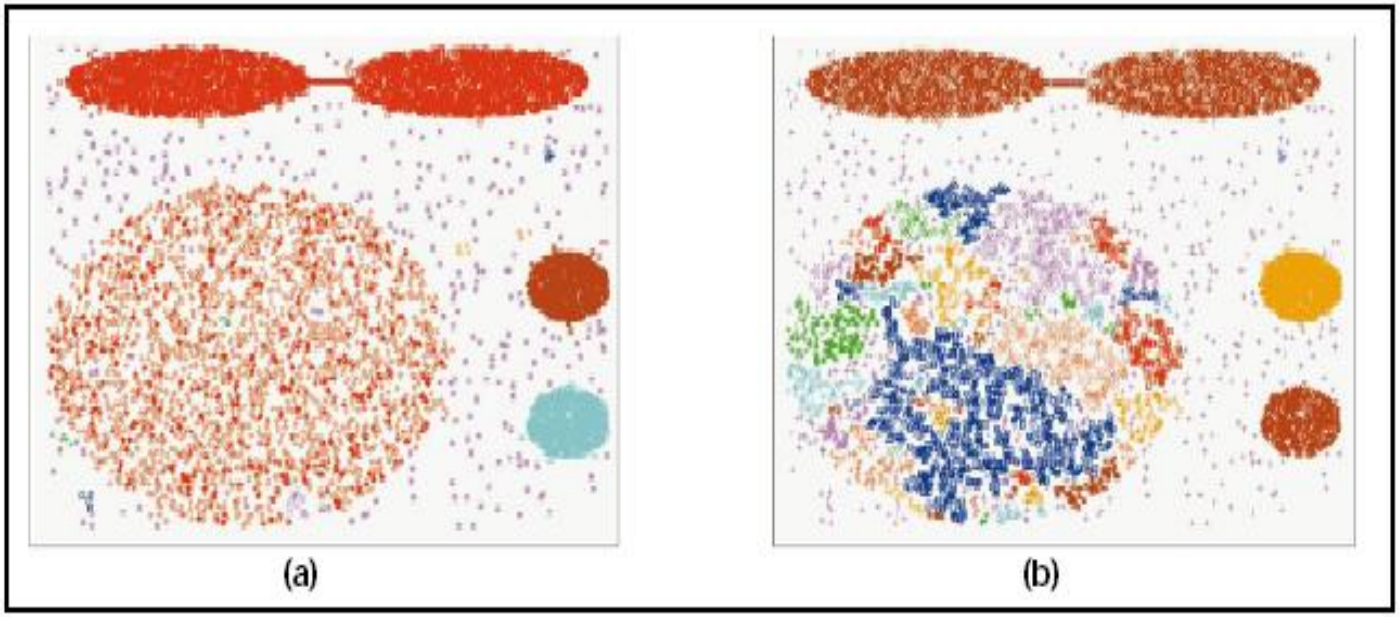
```
DBSCAN(D, eps, MinPts)
  C = 0
  for each unvisited point P in dataset D
    mark P as visited
    NeighborPts = regionQuery(P, eps)
    if sizeof(NeighborPts) < MinPts
      mark P as NOISE
    else
      C = next cluster
      expandCluster(P, NeighborPts, C, eps, MinPts)

expandCluster(P, NeighborPts, C, eps, MinPts)
  add P to cluster C
  for each point P' in NeighborPts
    if P' is not visited
      mark P' as visited
      NeighborPts' = regionQuery(P', eps)
      if sizeof(NeighborPts') >= MinPts
        NeighborPts = NeighborPts joined with NeighborPts'
    if P' is not yet member of any cluster
      add P' to cluster C

regionQuery(P, eps)
  return all points within P's eps-neighborhood (including P)
```

DBSCAN: Sensitive to Parameters

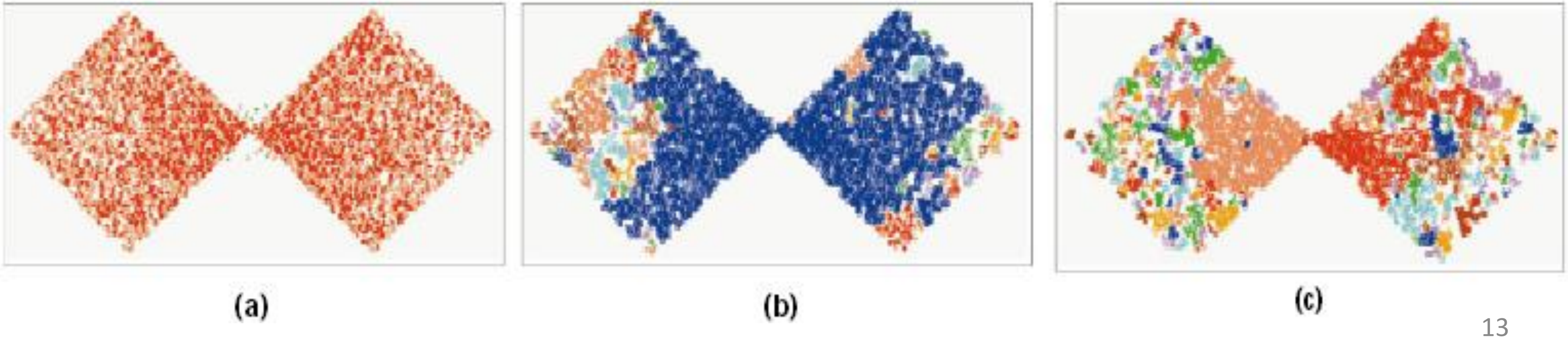
Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.



(a)

(b)

Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



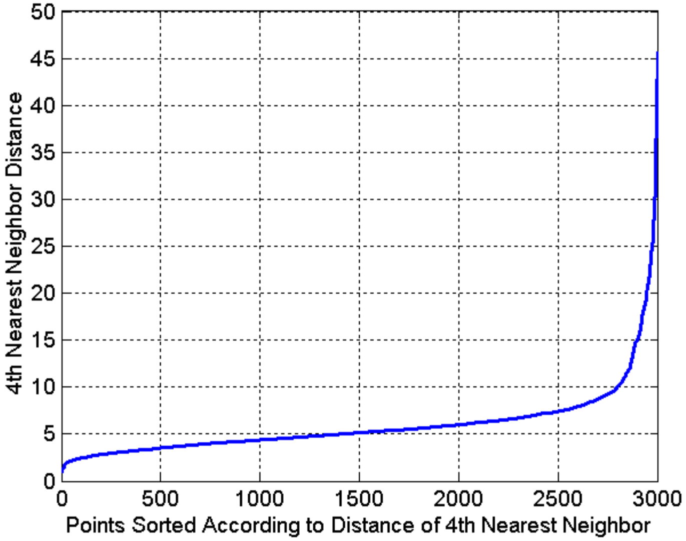
(a)

(b)

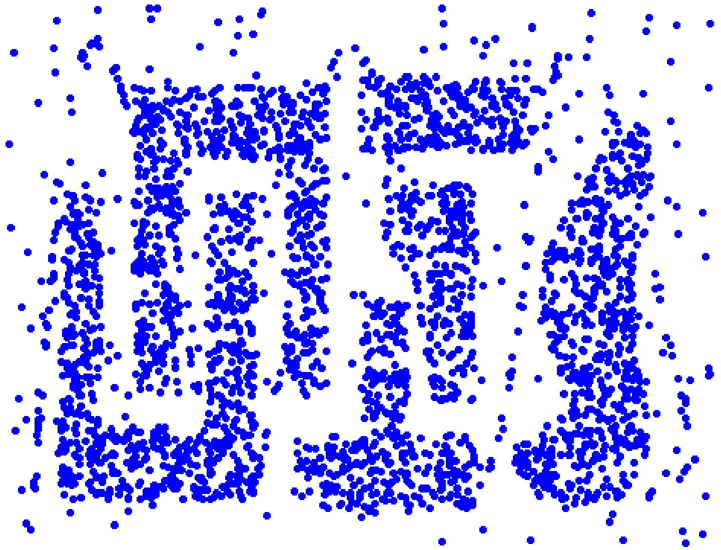
(c)

DBSCAN: Determining EPS and MinPts

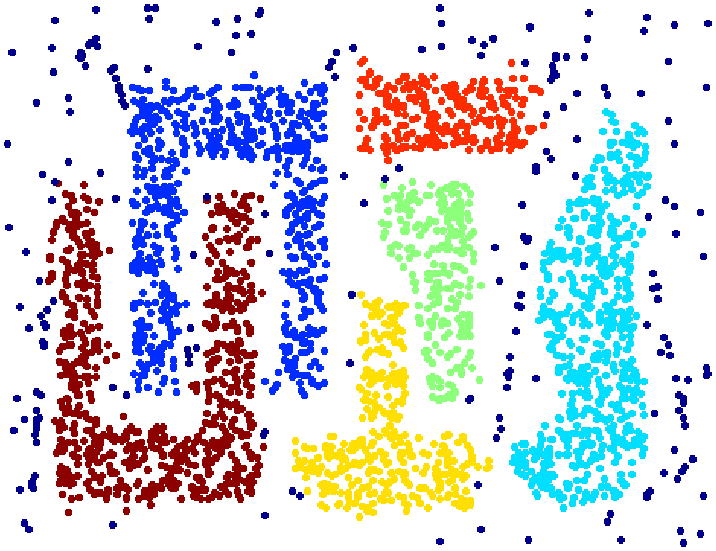
- Idea is that for points in a cluster, their kth nearest neighbors are at roughly the same distance
- Noise points have the kth nearest neighbor at farther distance
- So, plot sorted distance of every point to its kth nearest neighbor



When DBSCAN Works Well



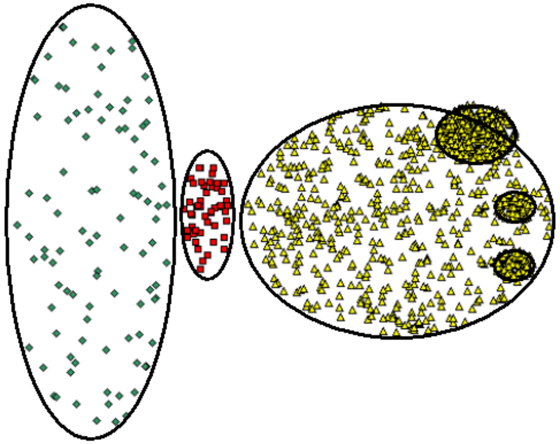
Original Points



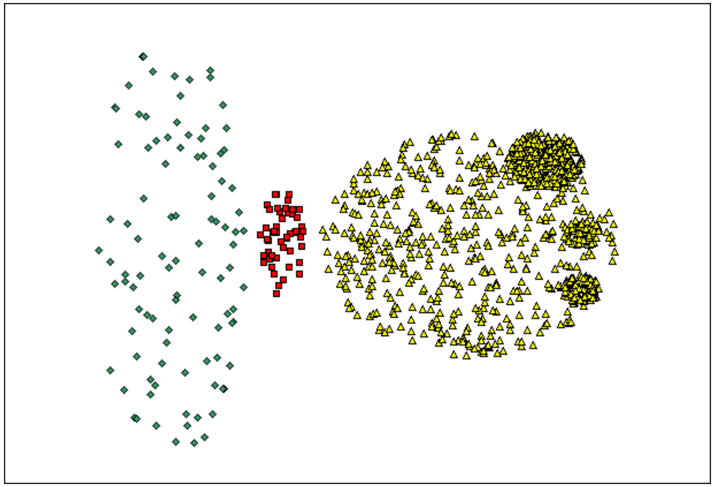
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

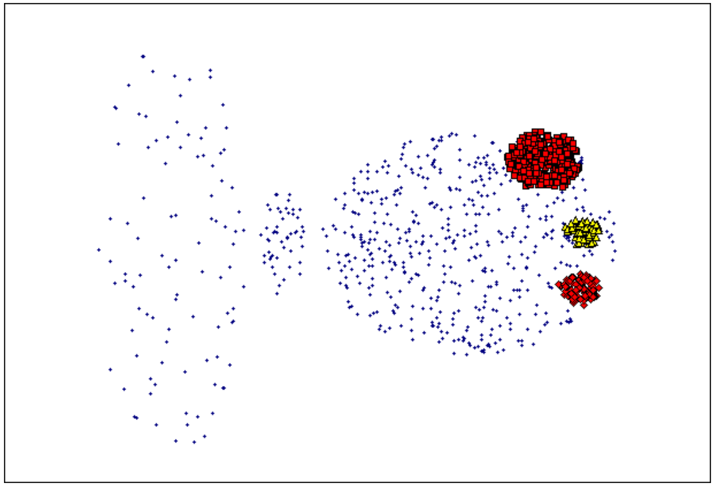
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

- **Cannot handle varying densities**
- **sensitive to parameters—hard to determine the correct set of parameters**

Take-away Message

- The basic idea of density-based clustering
- The two important parameters and the definitions of neighborhood and density in DBSCAN
- Core, border and outlier points
- DBSCAN algorithm
- DBSCAN's pros and cons